An Augmented Lagrangian Fish Swarm Based Method for Global Optimization

Ana Maria A.C. Rocha[*,a], Tiago F.M.C. Martins[b], Edite M.G.P. Fernandes[a]

[a] *Department of Production and Systems*
University of Minho, 4710-057 Braga, Portugal
[b] *Algorithm R & D Center, Portugal*

Abstract

This paper presents an augmented Lagrangian methodology with a stochastic population based algorithm for solving nonlinear constrained global optimization problems. The method approximately solves a sequence of simple bound global optimization subproblems using a fish swarm intelligent algorithm. A stochastic convergence analysis of the fish swarm iterative process is included. Numerical results with a benchmark set of problems are shown, including a comparison with other stochastic-type algorithms.

Key words: Augmented Lagrangian function, artificial fish swarm, stochastic convergence

1. Introduction

This paper presents a stochastic augmented Lagrangian methodology for solving continuous nonlinear constrained global optimization problems in the form:

$$\begin{array}{ll} \underset{x\in\Omega}{\text{minimize}} & f(x) \\ \text{subject to} & g_i(x) \leq 0\,, i = 1, \ldots, p \end{array} \tag{1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$, $g_i : \mathbb{R}^n \to \mathbb{R}$ are nonlinear continuous functions and $\Omega = \{x \in \mathbb{R}^n : l \leq x \leq u\}$. Any equality constraint $h(x) = 0$ is converted into an inequality one using $|h(x)| - \varepsilon \leq 0$, for a small fixed $\varepsilon > 0$. This is a common procedure in stochastic methods for global optimization. Since we do not assume convexity, problem (1) may have several minima and convergence to the global minimum is not guaranteed by gradient-based algorithms. Specially for medium- and large-scale problems, derivative-free and stochastic methods are promising alternatives. Some well-known derivative-free methods like the

[*] Corresponding author
Email addresses: arocha@dps.uminho.pt (Ana Maria A.C. Rocha), martins.tiago41@gmail.com (Tiago F.M.C. Martins), emgpf@dps.uminho.pt (Edite M.G.P. Fernandes)

deterministic pattern search method [23] cannot guarantee convergence to the global minimum. Stochastic algorithms based on a point-by-point search, like the filter-set-based simulated annealing [15], and on a population of points, like the particle swarm optimization [19, 29], genetic algorithm [3, 11, 36] and the electromagnetism-like mechanism [5] have been widely used in the past. Global optimization techniques are commonly used to solve important practical problems, such as those derived from the control of variance reduction techniques when applied to accelerators used in cancer therapy [14], the global kinetic modelling of the 2-chlorophenol oxidation in supercritical water [18] (a problem frequently found in waste waters), as well as the credit rating and the probability of default bucket problem [22] (that arises in the field of quantitative finance), and the analog integrated circuit design problem [28]. The Artificial Fish Swarm (AFS) algorithm is a recent and easy to implement artificial life computing algorithm that simulates fish swarm behaviors and has been successfully used in some engineering applications [20, 21, 37, 38]. A variety of techniques have been proposed to handle the equality and inequality constraints of the problem. The most widely used techniques rely on penalty functions. Here we are interested in a particular class of penalty functions known as augmented Lagrangian functions.

The area of Lagrange multiplier methods for constrained minimization has undergone a radical transformation starting with the introduction of augmented Lagrangian functions and methods of multipliers in 1968 by Hestenes [16] and Powell [30] in order to eliminate the duality gap between an equality constrained problem and its Lagrangian dual problem. Later, Rockafellar [32, 33] extended this method to deal with inequality constraints. Lagrangian dual methods have been serving as a fundamental solution methodology in convex programming. It is well known, however, that classical Lagrangian dual methods may fail to identify the optimal solution of the nonconvex problem (1) due to the existence of a duality gap. Since then, various modified augmented Lagrangian methods have been proposed. The strong duality properties and exact penalization of different types of augmented Lagrangians or nonlinear Lagrangians have been studied by many researchers (see for example, [4, 34]).

Mangasarian [27] analyzed the local convergence of a class of augmented Lagrangians that include Rockafellar's augmented Lagrangian as a special case. Global convergence of augmented Lagrangian methods for convex programming has been studied in [4, 32]. Convergence properties of the primal-dual methods based on Rockafellar and Wets' augmented Lagrangian function for inequality constrained global optimization problems can be seen in [25, 26]. Global convergence of the augmented Lagrangian method for nonconvex equality constrained problems was analyzed in [4, 30]. An indispensable assumption in most existing global convergence analysis for augmented Lagrangian methods is that the sequence of multiplier vectors generated in the algorithms is bounded. This restrictive assumption confines applications of augmented Lagrangian methods in many situations. Conn et al. [9], Conn, Gould, and Toint [10], and Lewis and Torczon [23] presented modified augmented Lagrangian methods for nonconvex optimization with equality constraints and proved global convergence

results without appealing to this assumption. Andreani et al. [1, 2], and Birgin, Castillo and Martínez [6] investigated the augmented Lagrangian methods for nonconvex constrained problems using safeguarding strategies of first-order Lagrange multiplier updates.

Constrained global optimization has been one of the challenging subjects in nonlinear optimization. On one hand, implementable methods for constrained global optimization have been developed only for some special problems such as concave minimization and monotone optimization. On the other hand, various deterministic and stochastic methods have been proposed for unconstrained global optimization (see, e.g., [8, 17]). In an augmented Lagrangian method a constrained global optimization problem is reduced into a sequence of unconstrained global optimization problems so that the methods developed for unconstrained global optimization can be used. Many deterministic global optimization methods rely on auxiliary functions to move from one local solution to another better one. These auxiliary functions aim to penalize a found local solution assigning heavy weights to it. This solution scheme was adopted in [41]. An alternative for escaping from a local minimizer of a constrained global optimization problem, is proposed in [39] where a filled function in unconstrained global optimization is combined with an idea of penalty function in constrained optimization. In [8] an augmented Lagrangian approach combined with a deterministic global optimization method (the αBB method) and its convex α-underestimation techniques is used.

The algorithm herein presented is a stochastic optimization method based on the augmented Lagrangian framework to solve constrained global optimization problems. To solve the bound constrained minimization subproblems, we propose a modified version of the AFS algorithm. This is the first attempt to integrate the AFS heuristic into an augmented Lagrangian framework. For completeness, we include a convergence analysis of the fish swarm algorithm. We present the condition that guarantees convergence of the fish swarm iterative process in mean square.

The remainder of this paper is organized as follows. Section 2 presents the main ideas concerned with the augmented Lagrangian framework, and Section 3 describes the proposed AFS algorithm and its convergence properties. In Section 4 we report our numerical experiments, including a comparison with other stochastic methods. Finally, Section 5 contains the conclusions and ideas for future work.

2. Augmented Lagrangian Framework

This section introduces a common constraint-handling method known as penalty technique. An augmented Lagrangian technique solves a sequence of very simple subproblems where the objective function penalizes all or some of the constraints violation. With most penalty functions, the solution of the constrained problem is reached for an infinite value of the penalty parameter. An augmented Lagrangian is a more sophisticated penalty function for which a

finite penalty parameter value is sufficient to yield convergence to the solution of the constrained problem [4].

This work has been motivated by other papers published on this subject, for example [6, 8, 23]. The herein used augmented Lagrangian function for solving problem (1) is the following:

$$\mathcal{L}_\rho(x,\mu) = f(x) + \frac{\rho}{2}\sum_{i=1}^{p}\left[\max\left(0,\, g_i(x) + \frac{\mu_i}{\rho}\right)\right]^2 \tag{2}$$

where $\mu = (\mu_1, \ldots, \mu_p)$ is the vector of Lagrange multipliers associated with $g_i(x) \leq 0, i = 1, \ldots, p$, and ρ is a positive penalty parameter. In this context, the corresponding subproblem is formulated as:

$$\underset{x\in\Omega}{\text{minimize}}\ \ \mathcal{L}_{\rho^k}(x,\mu^k) \tag{3}$$

for fixed values of ρ^k and μ^k. The problem of choosing the initial value for the penalty parameter is problematic. We use a proposal presented in recent work by Birgin, Floudas and Martínez [8],

$$\rho^1 = \max\left\{10^{-6}, \min\left\{10,\ \frac{2\,|f(x^0)|}{\|[g(x^0)]_+\|^2}\right\}\right\} \tag{4}$$

where x^0 is an arbitrary initial approximation and $[g(x)]_+ \in \mathbb{R}^p$ denotes the vector with components defined by $\max(0, g_i(x))$, $i = 1, \ldots, p$. Our updating of the penalty relies on a strategy that allows ρ to vary both upward and downward. Besides defining positive lower and upper bounds for the penalty updating, ρ^- and ρ^+ respectively, so that the subproblems (3) are maintained well conditioned, a fixed constant $\gamma > 1$ is used in the updating process. See Algorithm 1 below. The algorithm also updates the Lagrange multipliers using first order estimates and safeguarding schemes to maintain the sequence $\{\mu^k\}$ bounded, i.e., $\mu^k \in [0, \mu^+]$, for all k, where μ^+ is a sufficiently large positive constant.

The algorithm stops when a certain degree of constraints violation, measured by the norm of the vector ν^k, given by

$$\nu_i^k = \max\left\{g_i(x^k), -\frac{\mu_i^k}{\rho^k}\right\},\ i = 1, \ldots, p, \tag{5}$$

is satisfied, for a given tolerance ϵ^*; otherwise, the algorithm runs until a maximum of (outer) iterations, $k_{\max}$, is reached. We note that $\|\nu\|$ is also used to decide when the penalty parameter should be updated. If constraints violation improves, i.e., if $\|\nu^k\| \leq \alpha\|\nu^{k-1}\|$, with $0 < \alpha < 1$, then the penalty is maintained; otherwise we allow the penalty to increase in some cases, and decrease in others. When the degree of constraints violation, at iteration k, is under a certain tolerance ϵ^k then the penalization could be relaxed. Our suggestion is to decrease the penalty parameter (see Algorithm 1). The tolerance ϵ^k is also used to compute the approximate solution of subproblem (3) and is reduced as k increases, as follows:

$$\epsilon^k = \max\left\{\epsilon^*, 10^{-k}\right\}.$$

The algorithm AFS based on the augmented Lagrangian (AFS_aL) is presented below.

Algorithm 1. *AFS_aL Algorithm*

Given $\mu^+ > 0$, $0 < \epsilon^* \ll 1$, $0 < \alpha < 1$, $\gamma > 1$, $k_{\max}$, $0 < \rho^- < \rho^+$, $\mu^1 \in [0, \mu^+]$;

Step 1. *Randomly generate* x^0 *in* Ω;

Step 2. *Compute* ρ^1 *using* (4)*, and set* $k = 1$;

Step 3. *Repeat*

For a certain tolerance ϵ^k*, find an approximate minimizer* x^k *to the subproblem* (3) *using the AFS Algorithm*;
Update ν^k *using* (5);
If $k = 1$ *or* $\|\nu^k\| \leq \alpha \|\nu^{k-1}\|$ *then*
    $\rho^{k+1} = \rho^k$;
else
    if $\|\nu^k\| \leq \epsilon^k$ *then*
        $\rho^{k+1} = \max\{\rho^-, \frac{1}{\gamma}\rho^k\}$;
    else
        $\rho^{k+1} = \min\{\rho^+, \gamma\rho^k\}$;
    end if
end if
Update $\mu_i^{k+1} = \min\left\{\max\left\{0, \mu_i^k + \rho^k g_i(x^k)\right\}, \mu^+\right\}$, $i = 1, \ldots, p$;
Set $k = k + 1$;

Until $\|\nu^{k-1}\| \leq \epsilon^*$ *or* $k > k_{\max}$

The herein proposed technique for solving (3) uses a population-based algorithm that relies on swarm intelligence to converge towards the minimum value of the augmented Lagrangian function. This is the subject of the next section. Since the AFS algorithm provides a population of solutions, x^k is the best solution. We emphasize the importance of using x^k as one of the points of the population for the subproblem (3), at iteration $k + 1$. The remaining points of the population are randomly generated in the set Ω.

3. Artificial Fish Swarm Algorithm

In this section we present a stochastic population-based algorithm that simulates fish swarm behaviors to solve subproblem (3). This is an artificial life computing algorithm that has been used in some engineering context [20, 21, 37, 38]. We will use the words 'fish' and 'point' interchangeably throughout the paper. The artificial fish swarm algorithm is based on swarm intelligence and uses a population (or swarm) of points to identify promising regions looking for a global solution. For completeness, a condition that guarantees convergence of the fish

swarm iterative process is derived. The specific and used notation in the AFS algorithm is as follows: $x^i(t) \in \mathbb{R}^n$ denotes the ith point of the population at time/iteration t; $x^i_j(t) \in \mathbb{R}$ is the jth ($j = 1, \ldots, n$) component of the point x^i; and p_{size} is the number of points in the population.

3.1. Fish Swarm Behaviors

The fish swarm behaviors inside water are: (i) *random* behavior (in general, fish looks at random for food and other companion); (ii) *searching* behavior (when fish discovers a region with more food, it will go directly and quickly to that region); (iii) *swarming* behavior (when swimming, fish will swarm naturally in order to avoid danger); (iv) *chasing* behavior (when a fish in the swarm discovers food, the others will find the food dangling after it). The term food in the fish swarm system corresponds to a minimum in the optimization context.

The points in the population are evaluated using a fitness function. In this augmented Lagrangian framework the fitness function is $\mathcal{L}^k(x) \equiv \mathcal{L}_{\rho^k}(x, \mu^k)$.

A crucial parameter of the artificial fish swarm algorithm is a positive constant v that represents the ray of a closed neighborhood of x^i – the 'visual scope' – herein defined by

$$v = \delta \max_{j \in \{1,\ldots,n\}} (u_j - l_j),$$

where δ is a positive visual parameter that is reduced over the iterative process using the update formula $\delta = \max\{\delta_{\min}, \kappa_\delta \delta\}$, with $0 < \kappa_\delta < 1$, and $\delta_{\min} > 0$. Further, let I^i be the set of indices of the points inside the 'visual scope' of point x^i, where $i \notin I^i$ and $I^i \subset \{1, \ldots, p_{\text{size}}\}$, and let np^i be the number of points inside the 'visual scope'. Depending on the relative positions of the points inside the visual, the list of moves applied to each point x^i is the following:

When $np^i = 0$, the 'visual scope' is empty, and the point x^i, with no other points in its neighborhood to follow, moves randomly inside the visual searching for a better region.

When the 'visual scope' is considered crowded, i.e., when

$$T^i = \frac{np^i}{p_{\text{size}}} > \theta,$$

where $\theta \in (0, 1)$ is a crowd parameter, the point simulates the searching behavior. The point searches for a better region, choosing at random another point inside the visual, x^{rd}, and moves towards it if the new point improves over x^i; otherwise the point moves randomly inside the visual.

When the 'visual scope' is not crowded, the point is able either to chase moving towards the best point inside the visual, denoted by $x^{\min}$, or to swarm moving towards the central of the visual. First, if the condition

$$\mathcal{L}^k(x^{\min}) \equiv \min\left\{\mathcal{L}^k(x^j) : j \in I^i\right\} < \mathcal{L}^k(x^i)$$

is satisfied, x^i is moved towards $x^{\min}$; otherwise, the swarming behavior is tried.

The swarming behavior is characterized by a movement towards the central point of the 'visual scope' of x^i, defined by

$$\overline{x} = \frac{1}{np^i} \sum_{j \in I^i} x^j. \tag{6}$$

However, this movement is carried out only if the central point improves over x^i. If there is no improvement then the searching behavior is tried, as previously described.

The AFS algorithm is based on a set of trial moves mutually exclusive and sequentially simulated until a better position for each point is found. This can be simply described by the following iterative equation for the jth component $(j = 1, \ldots, n)$:

$$x_j^i = x_j^i + r \begin{cases} \left(x_j^{\min} - x_j^i\right), & \text{if } T^i \leq \theta \text{ and } np^i \neq 0 \text{ and } \mathcal{L}^k(x^{\min}) < \mathcal{L}^k(x^i) \\ \left(\overline{x}_j - x_j^i\right), & \text{if } T^i \leq \theta \text{ and } np^i \neq 0 \text{ and } \mathcal{L}^k(\overline{x}) < \mathcal{L}^k(x^i) \\ \left(x_j^{rd} - x_j^i\right), & \text{if } np^i \neq 0 \text{ and } \mathcal{L}^k(x^{rd}) < \mathcal{L}^k(x^i) \\ v, & \text{if } (np^i \neq 0 \text{ and } \mathcal{L}^k(x^{rd}) \geq \mathcal{L}^k(x^i)) \text{ or } np^i = 0 \end{cases}$$

where r is a random number uniformly distributed in $[0, 1]$. The bound constraints are enforced through a projection of the point x^i onto the set Ω, component by component given by:

$$x_j^i = \begin{cases} l_j, & \text{if} \quad x_j^i < l_j \\ x_j^i, & \text{if} \quad l_j \leq x_j^i \leq u_j \\ u_j, & \text{if} \quad x_j^i > u_j \end{cases} .$$

The AFS algorithm includes a procedure aiming to gather the local information around the best point of the population, the point with least fitness value

$$x^{\text{best}} = \arg\min \left\{ \mathcal{L}^k(x^i) : i = 1, \ldots, p_{\text{size}} \right\}.$$

It corresponds to a simple random line search applied component by component to x^{best}. The main steps are as follows. For each component j $(j = 1, \ldots, n)$, x^{best} is assigned to a temporary point y. Next, a random movement of length

$$\triangle \max_j (u_j - l_j), \;\; \triangle > 0 \tag{7}$$

is carried out and if a better point is obtained within $\max_{local}$ iterations, x^{best} is replaced by y, the search ends for that component and proceeds to another component.

This iterative process proceeds for a maximum of $l_{\max}$ iterations until the condition $(\mathcal{L}^k_{\text{avg}} - \mathcal{L}^k(x^{\text{best}})) \leq \epsilon^k$ is satisfied, where $\mathcal{L}^k_{\text{avg}}$ is the average fitness of all points in the population.

3.2. Fish Swarm Convergence in Mean Square

If we consider the position of a point from the population as a stochastic vector, the expectation and variance of the position, herein denoted by $\mathrm{E}[x^i(t)]$ and $\mathrm{V}[x^i(t)]$ respectively, can be calculated and the convergence property of the AFS algorithm can be analyzed. This is the first attempt to analyze the stochastic convergence of the AFS algorithm. In a stochastic context, the point population system is said to converge to P if for all $i = 1, \ldots, p_{\text{size}}$, $x^i(t)$ converges in mean square to P, i.e.,

$$\lim_{t\to\infty} \mathrm{E}[(x^i(t) - P)^2] = 0,$$

where P is a position in the search space. Noting that $\mathrm{E}[(x(t)-P)^2] = (\mathrm{E}[x(t)]-P)^2 + \mathrm{V}[x(t)]$, the convergence of $x^i(t)$ in mean square to P is equivalent to convergence of $\mathrm{E}[x^i(t)]$ to P and $\mathrm{V}[x^i(t)]$ to 0 simultaneously. The values of $x^{\min}(t)$, $\overline{x}(t)$, $x^{rd}(t)$ and $v(t)$ vary throughout the iterative process. However, for this analysis we assume that they are kept constant for a set of iterations. Thus, all points move independently and, for any i, only the point i needs to be analyzed. For simplicity, we consider one-dimensional vectors. For the convergence purpose, we rewrite the iterative equation in the form

$$\begin{aligned} x^i(t+1) = \quad & x^i(t) + r(t)\left(c_1(x^{\min}(t) - x^i(t)) + c_2(\overline{x}(t) - x^i(t))\right. \\ & \left. + c_3(x^{rd}(t) - x^i(t)) + c_4 v(t)\right) \end{aligned} \tag{8}$$

where t represents the iteration counter and c_1, c_2, c_3, c_4 are integer parameters from the set $\{0, 1\}$ subject to $c_1 + c_2 + c_3 + c_4 = 1$. Using (8), the following non-homogeneous recurrence relation is obtained

$$x(t+1) = x(t)\left(1 - r(t)(c_1 + c_2 + c_3)\right) + r(t)(c_1 x^{\min} + c_2\overline{x} + c_3 x^{rd} + c_4 v). \tag{9}$$

Since $x(0)$ and $r(t)$ are random numbers, each $x(t)$ is a random variable and the iterative process $\{x(t)\}$ is a stochastic process.

For the convergence study, we first analyze the convergence of $\mathrm{E}[x(t)]$. According to (9), the iteration equation of $\{\mathrm{E}[x(t)]\}$ is

$$\begin{aligned} \mathrm{E}[x(t+1)] = \quad & \mathrm{E}[x(t)]\left(1 - \mathrm{E}[r(t)](c_1 + c_2 + c_3)\right) \\ & + \mathrm{E}[r(t)]\left(c_1 x^{\min} + c_2\overline{x} + c_3 x^{rd} + c_4 v\right) \\ = \quad & \mathrm{E}[x(t)]\left(1 - \frac{c_1 + c_2 + c_3}{2}\right) + \frac{c_1 x^{\min} + c_2\overline{x} + c_3 x^{rd} + c_4 v}{2}. \end{aligned}$$

noting that $x(t)$ is independent on $r(t)$ and $\mathrm{E}[r(t)] = \frac{1}{2}$. The characteristic equation of this iterative process is $\lambda - 1 + \frac{c_1+c_2+c_3}{2} = 0$.

Theorem 1. *Given $c_1, c_2, c_3, c_4 \in \{0, 1\}$ such that $c_1+c_2+c_3+c_4 = 1$, then the iterative process $\{\mathrm{E}[x(t)]\}$ converges to $(c_1 x^{\min} + c_2\overline{x} + c_3 x^{rd} + c_4 v)/(c_1 + c_2 + c_3)$ if and only if $c_1 + c_2 + c_3 = 1$.*

Proof. From $c_1 + c_2 + c_3 + c_4 = 1$, the sum $c_1 + c_2 + c_3$ is either 0 or 1. The convergence condition of the iterative process $\{\mathrm{E}[x(t)]\}$ is that the absolute value of the characteristic equation root is less than 1. Since the root is

$$\lambda = 1 - \frac{c_1 + c_2 + c_3}{2},$$

then unique condition that guarantees convergence of $\{\mathrm{E}[x(t)]\}$ is $c_1+c_2+c_3=1$. The convergent value $\mathrm{E}[x]$ can be obtained by using

$$\mathrm{E}[x] = \mathrm{E}[x]\left(1-\frac{c_1+c_2+c_3}{2}\right)+\frac{c_1x^{\min}+c_2\overline{x}+c_3x^{rd}+c_4v}{2}$$

which yields $\mathrm{E}[x] = (c_1x^{\min}+c_2\overline{x}+c_3x^{rd}+c_4v)/(c_1+c_2+c_3)$. ■

We now analyze the convergence of $\mathrm{V}[x(t)]$. Let $Z = c_1x^{\min}+c_2\overline{x}+c_3x^{rd}+c_4v$ and $\mu = (c_1x^{\min}+c_2\overline{x}+c_3x^{rd}+c_4v)/(c_1+c_2+c_3)$. For convenience, we define a new random variable $y(t) = x(t)-\mu$, and obviously $\mathrm{E}[y(t)] = \mathrm{E}[x(t)]-\mu$ and $\mathrm{V}[y(t)] = \mathrm{V}[x(t)]$. From (9) we then get

$$\begin{array}{rcl} y(t+1) = & y(t)-r(t)(c_1+c_2+c_3)(x(t)-\mu)-r(t)(c_1+c_2+c_3)\mu+r(t)Z \\ = & y(t)\left(1-r(t)(c_1+c_2+c_3)\right)-r(t)\left((c_1+c_2+c_3)\mu-Z\right) \\ = & y(t)\left(1-r(t)(c_1+c_2+c_3)\right) \end{array} . \tag{10}$$

Theorem 2. *Given $c_1,c_2,c_3,c_4\in\{0,1\}$ such that $c_1+c_2+c_3+c_4=1$, then the iterative process $\{\mathrm{V}[x(t)]\}$ converges to 0 if and only if $c_1+c_2+c_3=1$.*

Proof. Using (10), we obtain the iteration equation of $\mathrm{V}[y(t)]$:

$$\begin{array}{rl} \mathrm{V}[y(t+1)] = & \left(\mathrm{E}[y(t)]\right)^2\,\mathrm{V}[(1-r(t)(c_1+c_2+c_3))] \\ & +\mathrm{V}[y(t)]\,\left(\mathrm{E}[(1-r(t)(c_1+c_2+c_3))]\right)^2 \\ & +\mathrm{V}[y(t)]\,\mathrm{V}[(1-r(t)(c_1+c_2+c_3))] \end{array}$$

using $\mathrm{V}[R_1R_2] = (\mathrm{E}[R_1])^2\mathrm{V}[R_2]+\mathrm{V}[R_1](\mathrm{E}[R_2])^2+\mathrm{V}[R_1]\mathrm{V}[R_2]$ for any two independent random variables R_1 and R_2. Since $\mathrm{V}[r(t)] = \frac{1}{12}$, then

$$\begin{array}{rl} \mathrm{V}[y(t+1)] = & (\mathrm{E}[y(t)])^2\,\dfrac{(c_1+c_2+c_3)^2}{12}+\mathrm{V}[y(t)]\,\left(1-\dfrac{c_1+c_2+c_3}{2}\right)^2 \\ & +\mathrm{V}[y(t)]\,\dfrac{(c_1+c_2+c_3)^2}{12}. \end{array}$$

Using previously defined relations, we obtain the iteration equation of $\mathrm{V}[x(t)]$:

$$\begin{array}{rl} \mathrm{V}[x(t+1)] = & \mathrm{V}[x(t)]\left(1-\dfrac{c_1+c_2+c_3}{2}\right)^2+\mathrm{V}[x(t)]\dfrac{(c_1+c_2+c_3)^2}{12} \\ & +(\mathrm{E}[x(t)]-\mu)^2\dfrac{(c_1+c_2+c_3)^2}{12} \end{array} \tag{11}$$

and the corresponding characteristic equation is

$$\lambda-\left(1-\frac{c_1+c_2+c_3}{2}\right)^2-\frac{(c_1+c_2+c_3)^2}{12}=0.$$

Since equation (11) depends on $\mathrm{E}[x(t)]$, convergence condition in Theorem 1 should also be satisfied. Since the root of the characteristic equation must satisfy

$$\left|1-(c_1+c_2+c_3)+\frac{(c_1+c_2+c_3)^2}{3}\right|<1,$$

we conclude that condition $c_1+c_2+c_3=1$ guarantees convergence of $\{\mathrm{V}[x(t)]\}$.

The convergent value is obtained from (11) as follows:

$$\mathrm{V}[x]=\mathrm{V}[x]\left(\left(1-\frac{c_1+c_2+c_3}{2}\right)^2+\frac{(c_1+c_2+c_3)^2}{12}\right)+(\mathrm{E}[x]-\mu)^2\frac{(c_1+c_2+c_3)^2}{12}$$

which gives $\mathrm{V}[x]=0$. ■

4. Numerical Experiments

In this section, we report the results of our numerical study, after running a set of 30 benchmark constrained global problems: g01 to g24 (g02, g03, g08 and g12 are maximization problems) described in [24], and Problem 1, Problem 2 (a),(b),(c),(d) and Problem 3 in [7]. The algorithm was coded in C#, and the results were obtained in a computer Core 2 Duo 2.0 GHz, with 2 GB 667MHz, running Microsoft Windows XP SP2.

Since the algorithm relies on some random parameters and variables, we solve each problem 10 times and take average of the obtained solutions, herein denoted by f_{avg}. The best of the solutions found after all runs is denoted by f_{best}. Our test set contains problems of different dimensions, from $n=2$ to $n=24$. Thus, the size of the population depends on n, with an upper bound to reduce the overall computational effort: $p_{\mathrm{size}}=\min\{10n,200\}$. Some of the fixed parameters are set in this study as follows: $\rho^-=10^{-12}$, $\rho^+=10^{12}$, $\mu^+=10^{12}$, $\epsilon^*=10^{-12}$, and $\gamma=10$, $\alpha=0.5$ as proposed in [8]. The initial multiplier vector is set to the null vector. Several tests were done in order to choose appropriate values for the parameters. Common values for the parameter ε vary from 10^{-3} to 10^{-6} [15, 24, 29, 31]. We observed that setting $\varepsilon=10^{-5}$ gave a slightly better performance than with the other values. In the AFS algorithm, we set the initial δ to 1, and $\kappa_\delta=0.9$, $\delta_{\min}=10^{-8}$, $\theta=0.8$. According to the sensitivity study carried out in [13], these values led to good accuracy solutions at a reasonable computational cost. Since the local search is similar to that proposed in [5], we use the values therein suggested, i.e., $\triangle=0.001$ in (7) and $\max_{local}=10$. The chosen values for the upper bounds of the penalty parameter and Lagrange multipliers have no significant effect on the performance of the algorithm as long as they are sufficiently large. The same is true for the lower bounds. The parameters γ, $l_{\max}$ and $k_{\max}$ have been subject to a sensitivity study as shown below.

4.1. Sensitivity analysis of some parameters

To analyze the effect of some parameters in the performance of the proposed augmented Lagrangian algorithm, we use the Dolan and Moré's [12] performance profiles. The presented profiles are based on the metrics: f_{avg} and f_{best}. If $\mathcal{P}$ and $\mathcal{S}$ correspond to the set of problems and the set of solvers in comparison, respectively, and $m_{p,s}$ is the value of the metric obtained when solving problem $p \in \mathcal{P}$ by solver $s \in \mathcal{S}$, then the comparison uses the following performance ratio:

$$r_{p,s} = \begin{cases} 1 + m_{p,s} - \min\{m_{p,s} : s \in \mathcal{S}\}, \text{ if } \min\{m_{p,s} : s \in \mathcal{S}\} < \beta \\ \frac{m_{p,s}}{\min\{m_{p,s} : s \in \mathcal{S}\}}, \text{ otherwise} \end{cases},$$

where β is a small positive parameter. The overall performance of the solver s is assessed by the probability (for $s \in \mathcal{S}$) that $r_{p,s}$ is within a factor $\tau \in \mathbb{R}$ of the best possible ratio, i.e., by $\rho_s(\tau) =$ (no. of problems where $r_{p,s} \leq \tau$)/(total no. of problems). The value of $\rho_s(1)$ gives the probability that the solver s will win over the others in the set, and for large values of τ, the $\rho_s(\tau)$ measures the solver robustness. The higher the ρ_s the better the solver is.

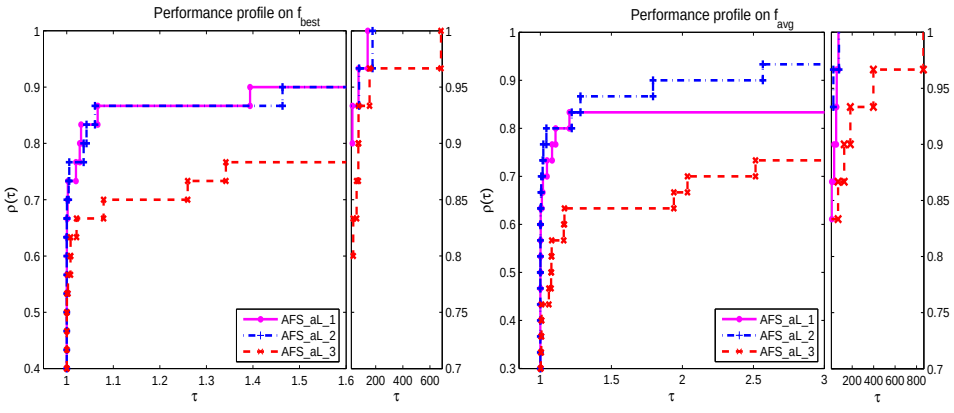

Figure 1: Performance profiles on f_{best} and f_{avg}

Table 1: Average number of function evaluations

AFS_aL_1	AFS_aL_2	AFS_aL_3
218125	217164	208196

Figure 1 contains two sets of profiles to compare the effects of three different parameter set values in the AFS augmented Lagrangian algorithm, that we refer for simplicity by: AFS_aL_1 (with $\gamma = 10$ in ρ update, $k_{\max} = 20$ and $l_{\max} = \max\{50, 10n\}$); AFS_aL_2 (with $\gamma = 2$ in ρ update, $k_{\max} = 20$ and

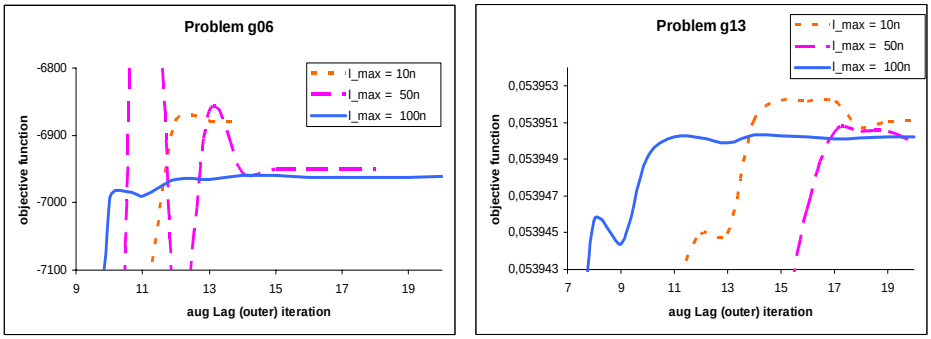

Figure 2: Comparison of convergence histories, for problems g06 and g13

$l_{\max} = \max\{50, 10n\}$); AFS_aL_3 (with $\gamma = 10$ in ρ update, $k_{\max} = 100$ and $l_{\max} = \max\{20, 2n\}$). The plots on the left represent the profiles of f_{best} and the ones on the right show the profiles of f_{avg}. From the profiles we may conclude that the most efficient version is AFS_aL_2 followed by AFS_aL_1 and then by AFS_aL_3. Decreasing the number of (AFS) inner iterations while increasing the number of outer (augmented Lagrangian) iterations did not improve solutions accuracy. We report in Table 1 the average number of function evaluations required by each of the three versions in comparison. From Figure 1 and Table 1 we conclude that reducing the number of allowed outer iterations and at the same time increasing the number of inner iterations (of the AFS algorithm) yields more accurate solutions, although at a cost of more function evaluations (about 10000 evaluations).

To further analyze the effect of the maximum number of AFS algorithm iterations on the performance of the algorithm, we selected two small problems. g06 has a cubic objective function with $n = 2$, two nonlinear inequality constraints, and the size of the feasible region has been estimated as 0.0066% [24]. Its optimum solution is -6961.814. g13 has an exponential objective function with $n = 5$ and three nonlinear equality constraints. The reported size of the feasible region is 0.0000% and the solution is 0.05395. We plot in Figure 2 the best function value obtained at each outer iteration. The 'solid', 'dash' and 'dot' lines show the convergence histories using different $l_{\max}$ values: $l_{\max} = 100n$, $l_{\max} = 50n$ and $l_{\max} = 10n$ respectively. In all cases we set $k_{\max} = 20$. Clearly $l_{\max} = 100n$ (the solid line) leads to a convergence that is steady and approaches the neighborhood of the solution more quickly.

4.2. Effect of population size on the algorithm

To analyze the effect of population size on the algorithm, problems g04, g09, g14 and g19 were selected and run with three different values of p_{size}. In these experiments, we choose $10n$, $20n$ and $30n$. The results concerning the best function value, the standard deviation ('st. dev.') of the function values, over the 10 runs, and the average number of outer iterations ('k_{avg}') are listed in

Table 2: Effect of population size (with $k_{\max} = 20$ and $l_{\max} = \max\{50, 10n\}$)

Problem	n	p_{size}	f_{best}	f_{avg}	st. dev.	k_{avg}
g04	5	$10n$	-30665.53829	-30665.50434	0.0234	19
		$20n$	-30665.53743	-30665.51768	0.0130	18
		$30n$	-30665.53158	-30665.51623	0.0153	19
g09	7	$10n$	680.630079	681.717354	1.7644	14
		$20n$	680.628487	680.702862	0.2155	17
		$30n$	680.630066	680.657272	0.0801	18
g14	10	$10n$	-47.674715	-47.323734	0.3273	19
		$20n$	-47.673703	-47.563525	0.1136	20
		$30n$	-47.742806	-47.508858	0.1479	20
g19	15	$10n$	32.848561	38.810566	5.2191	17
		$20n$	33.221027	38.544367	4.9437	15
		$30n$	32.850945	36.636361	5.6039	18

Table 2. We may conclude that efficiency and consistency of the algorithm do not depend too much on the population size.

4.3. Comparison with other stochastic algorithms

We compare our results with those in [3, 15, 31, 40]. In [3], an adaptive penalty technique is implemented within a genetic algorithm. The authors in [15] use a filter-set-based procedure in a simulated annealing method. A particle swarm optimization (PSO) combined with a constraint-handling mechanism is proposed in [31] and in [40], a hybrid PSO with Nelder-Mead simplex search is proposed. The results reported in the Table 3 were taken from the original references. Our comparison only includes problems g01-g13 since this is the set that mostly appears in the literature. The table lists the known reference solutions, f^*, as reported in [24], the best obtained function value after the runs, f_{best}, the average of the obtained function values, f_{avg}, and the average number of function evaluations, $nfeval_{\text{avg}}$. The character '-' means that the result is not available in the paper. We gather that the minimum number of function evaluations in [3] is 100000, since 1000 iterations with a population of 100 points are implemented. Our results are obtained with $k_{\max} = 20$, $l_{\max} = \max\{50, 10n\}$ and use a population of $\min\{10n, 200\}$. They are quite satisfactory except for problem g02. Solution consistency seems adequate regarding the reduced number of function evaluations required. Problems g01, g02 and g03 are the exceptions. In terms of computational costs (number of function evaluations) our approach wins over the others in problems g07, g09, g10 and g13. As far as the best and/or average function values are concerned, our results for g04, g07, g09, g10 and g13 attain solutions nearer the optimal than most of the other methods in comparison. The results in Table 3 seem to show that the algorithm in [40] outperforms the others in comparison. This is

expected since the algorithm uses gradient information derived from the constraint set in order to direct infeasible solutions towards the feasible region. All the others are derivative-free techniques.

5. Conclusions

A stochastic augmented Lagrangian methodology has been presented in this paper. The main issues addressed in the paper include the integration of the artificial fish swarm heuristic into an augmented Lagrangian framework, the stochastic convergence analysis of the fish swarm algorithm, and the practical analysis of the effect of some parameters on the performance of the algorithm. Comparison with other stochastic-type methods allows us to conclude that the herein proposed augmented Lagrangian fish swarm based method is mostly able to converge to the solutions with a specified tolerance and is not expensive in terms of function evaluations. Despite using the methodology of converting equality constraints into inequality ones, the results show that our approach is competitive in the sense that good accuracy solutions are obtained with a reasonable computational effort. We remark that the herein implemented AFS algorithm does not seem suitable for parallelization aiming to reduce significantly the CPU time, since each point moves along one direction at a time. Our next step concerning this stochastic augmented Lagrangian paradigm is to include a separate penalty term into the augmented Lagrangian function for equality constraint-handling. Then, the extended augmented Lagrangian approach will also be tested in the solution of some engineering practical problems.

Acknowledgments

The authors wish to thank four anonymous referees for their careful reading of the manuscript and their comments and suggestions.

References

[1] R. Andreani, E. G. Birgin, J. M. Martínez, and M. L. Schuverdt, On Augmented Lagrangian Methods with General Lower-Level Constraints, SIAM Journal on Optimization, 18 (2007) 1286–309.

[2] R. Andreani, E. G. Birgin, J. M. Martínez, and M. L. Schuverdt, Augmented Lagrangian Methods Under the Constant Positive Linear Dependence Constraint Qualification, Mathematical Programming, 111 (2008) 5–32.

[3] H.J.C. Barbosa, A.C.C. Lemonge, An adaptive penalty method for genetic algorithms in constrained optimization problems, in *Frontiers in Evolutionary Robotics*, H. Iba (ed.) (2008) (ISBN: 978-3-902613-19-6) I-Tech Education Publ., Austria.

Table 3: Comparison of results

	f^*		AFS_aL	[3]	[15]	[31]	[40]
g01	-15.0000	f_{best}	-14.9994	-14.9998	-14.9991	-15.0000	-15.0000
		f_{avg}	-14.8818	-14.9989	-14.9933	-15.0000	-15.0000
		$nfeval_{\text{avg}}$	339900	-	205748	340000	41959
g02	0.80362	f_{best}	0.59393	0.79252	0.75491	0.80343	0.80362
		f_{avg}	0.47966	0.72555	0.37171	0.79041	0.79643
		$nfeval_{\text{avg}}$	889416	-	227832	340000	2229858
g03	1.00000	f_{best}	0.99858	0.99725	1.00000	1.00472	1.00000
		f_{avg}	0.99192	0.77797	0.99919	1.00381	1.00000
		$nfeval_{\text{avg}}$	233264	-	314938	340000	64108
g04	-30665.54	f_{best}	-30665.54	-30665.32	-30665.54	-30665.50	-30665.54
		f_{avg}	-30665.50	-30578.55	-30665.47	-30665.50	-30665.54
		$nfeval_{\text{avg}}$	53773	-	86154	340000	19658
g05	5126.498	f_{best}	5126.681	5126.779	5126.498	5126.640	5126.359
		f_{avg}	5134.745	5323.866	5126.498	5461.081	5126.359
		$nfeval_{\text{avg}}$	45196	-	47661	340000	25253
g06	-6961.814	f_{best}	-6961.640	-6961.448	-6961.814	-6961.810	-6961.824
		f_{avg}	-6851.709	-6805.229	-6961.814	-6961.810	-6961.824
		$nfeval_{\text{avg}}$	17080	-	44538	340000	9856
g07	24.3062	f_{best}	24.3065	24.5450	24.3106	24.3511	24.3062
		f_{avg}	24.3109	27.8486	24.3795	24.3558	24.4883
		$nfeval_{\text{avg}}$	218989	-	404501	340000	1129252
g08	0.09583	f_{best}	0.09583	0.09583	0.09583	0.09583	0.09583
		f_{avg}	0.09583	0.08769	0.09583	0.09583	0.09583
		$nfeval_{\text{avg}}$	13987	-	56476	340000	2103
g09	680.630	f_{best}	680.630	680.681	680.630	680.638	680.630
		f_{avg}	680.631	681.470	680.636	680.852	680.630
		$nfeval_{\text{avg}}$	99198	-	324569	340000	422498
g10	7049.25	f_{best}	7055.47	7070.56	7059.86	7057.59	7049.30
		f_{avg}	7134.54	8063.29	7509.32	7560.05	7049.57
		$nfeval_{\text{avg}}$	140395	-	243520	340000	881161
g11	0.75000	f_{best}	0.74999	0.75217	0.75000	0.75000	0.75000
		f_{avg}	0.74999	0.88793	0.75000	0.75011	0.75000
		$nfeval_{\text{avg}}$	22220	-	23722	340000	743
g12	1.00000	f_{best}	1.00000	-	1.00000	1.00000	1.00000
		f_{avg}	0.99808	-	1.00000	1.00000	1.00000
		$nfeval_{\text{avg}}$	2204	-	59355	340000	923
g13	0.05395	f_{best}	0.05395	-	0.05395	0.06867	0.05395
		f_{avg}	0.05885	-	0.29772	1.71643	0.05485
		$nfeval_{\text{avg}}$	58204	-	120268	340000	265548

[4] D.P. Bertsekas, Nonlinear Programming, 2nd edn. Athena Scientific, Belmont, 1999.

[5] S.I. Birbil, S. Fang, R. Sheu, On the convergence of a population-based global optimization algorithm, Journal of Global Optimization, 30 (2004) 301–318.

[6] E. G. Birgin, R. A. Castillo, and J. M. Martinez, Numerical comparison of augmented Lagrangian algorithms for nonconvex problems, Computational Optimization and Applications, 31 (2005) 31–55.

[7] E.G. Birgin, C.A. Floudas, J.M. Martinez, Global minimization using an Augmented Lagrangian method with variable lower-level constraints, Technical Report MCDO121206 January 2007 (www.ime.usp.br/ egbirgin/publications/bfmreport.pdf).

[8] E.G. Birgin, C.A. Floudas, J.M. Martinez, Global minimization using an Augmented Lagrangian method with variable lower-level constraints, Mathematical Programming, Ser. A, (2009) DOI:10.1007/s10107-009-0264-y.

[9] A. R. Conn, N. I. M. Gould, A. Sartenaer, and Ph. L. Toint, Convergence properties of an augmented Lagrangian algorithm for optimization with a combination of general equality and linear constraints, SIAM Journal on Optimization, 6 (1996) 674–703.

[10] A. R. Conn, N. I. M. Gould, and Ph. L. Toint, A globally convergent augmented Lagrangian algorithm for optimization with general constraints and simple bounds, SIAM Journal on Numerical Analysis, 28 (1991) 545–572.

[11] K. Deep, Dipti, A self-organizing migrating genetic algorithm for constrained optimization, Applied Mathematics and Computation, 198 (2008) 237–250.

[12] E.D. Dolan, J.J. Moré, Benchmarking optimization software with performance profiles, Mathematical Programming, 91 (2002) 201–213.

[13] E.M.G.P. Fernandes, T.F.M.C. Martins, A.M.A.C. Rocha, Fish swarm intelligent algorithm for bound constrained global optimization, Proc. of the 2009 CMMSE, J.V. Aguiar (ed.), ISBN: 978-84-612-9727-6 (2009) 461–472.

[14] S.Garca-Pareja, M. Vilches, A.M. Lallena, Ant colony method to control variance reduction techniques in the Monte Carlo simulation of clinical electron linear accelerators of use in cancer therapy, Journal of Computational and Applied Mathematics, 233 (2010) 1534–1541.

[15] A.-R. Hedar, M. Fukushima, Heuristic pattern search and its hybridization with simulated annealing for nonlinear global optimization, Optimization Methods and Software, 19 (2004) 291–308.

[16] M. R. Hestenes, Multiplier and gradient methods, Journal of Optimization Theory and Applications, 4 (1969) 303–320.

[17] R. Horst and H. Tuy, Global Optimization: Deterministic Approaches, 3rd ed., Springer–Verlag, Berlin, 1996.

[18] C. Hu and X. Yan, A novel adaptive differential evolution algorithm with application to estimate kinetic parameters of oxidation in supercritical water, Engineering Optimization, 41 (2009) 1051–1062.

[19] M. Jiang, Y.P. Luo, S.Y. Yang, Stochastic convergence analysis and parameter selection of the standard particle swarm optimization algorithm, Information Processing Letters, 102 (2007) 8–16.

[20] M. Jiang, Y. Wang, S. Pfletschinger, M.A. Lagunas, D. Yuan, Optimal multiuser detection with artificial fish swarm algorithm, CCIS 2, ICIC 2007, D.-S. Huang, L. Heutte and M. Loog (eds.), Springer-Verlag, (2007) 1084–1093.

[21] M. Jiang, N. Mastorakis, D. Yuan, M.A. Lagunas, Image segmentation with improved artificial fish swarm algorithm, Lecture Notes in Electrical Engineering, 28, Proceedings of ECC, N. Mastorakis, V. Mladenov, V.T. Kontargyri (Eds.), ISBN: 978-0-387-84818-1, Springer-Verlag (2009) 133–138.

[22] T. Krink, S. Paterlini, A. Resti, Using differential evolution to improve the accuracy of bank rating systems, Computational Statistics & Data Analysis, 52 (2007) 68–87.

[23] R. M. Lewis and V. Torczon, A globally convergent augmented Lagrangian pattern search algorithm for optimization with general constraints and simple bounds, SIAM Journal on Optimization, 12 (2002) 1075–1089.

[24] J.J. Liang, T.P. Runarsson, E. Mezura-Montes, M. Clerc, P.N. Suganthan, C.A.C. Coello, K. Deb, Problem definitions and evaluation criteria for the CEC2006 special session on constrained real-parameter optimization 2006. (http://www.ntu.edu.sg/home/EPNSugan/index_files/CEC-06/CEC06.htm)

[25] H. Luo, X. Sun, H. Wu, On the convergence of augmented Lagrangian methods for constrained global optimization, SIAM Journal on Optimization, 18(4) (2007) 1209–1230.

[26] H. Luo, X. Sun, H. Wu, Convergence properties of augmented Lagrangian methods for constrained global optimization, Optimization Methods and Software, 23 (2008) 763–778.

[27] O. L. Mangasarian, Unconstrained Lagrangians in nonlinear programming, SIAM Journal on Control and Optimization, 13 (1975) 772–791.

[28] J. Olensek, A. Burmen, J. Puhan, T. Tuma, DESA: a new hybrid global optimization method and its application to analog integrated circuit sizing, Journal of Global Optimization, 44 (2009) 53–77.

[29] Y.G. Petalas, K.E. Parsopoulos, M.N. Vrahatis, Memetic particle swarm optimization, Annals of Operations Research, 156 (2007) 99–127.

[30] M. J. D. Powell, A method for nonlinear constraints in minimization problems, in Optimization, R. Fletcher (ed.), London and New York, Academic Press, (1969) 283–298.

[31] G.T. Pulido, C.A.C. Coello, A constrained-handling mechanism for particle swarm optimization, Congress on Evolutionary Computation, IEEE Vol 2 (2004) 1396–1403.

[32] R.T. Rockafellar, The multiplier method of Hestenes and Powell applied to convex programming, Journal of Optimization Theory and Applications, 12 (1973) 555–562.

[33] R.T. Rockafellar, Augmented Lagrange multiplier functions and duality in nonconvex programming, SIAM Journal on Control and Optimization, 12 (1974) 268–285.

[34] R.T. Rockafellar, Lagrange multipliers and optimality, SIAM Review, 35 (1993) 183–238.

[35] M.-J. Tahk, H.-W. Woo, M.-S. Park, A hybrid optimization method of evolutionary and gradient search, Engineering Optimization, 39 (2007) 87–104.

[36] I.G. Tsoulos, Solving constrained optimization problems using a novel genetic algorithm, Applied Mathematics and Computation, 208 (2009) 273–283.

[37] C.-R. Wang, C.-L. Zhou, J.-W. Ma, An improved artificial fish-swarm algorithm and its application in feed-forward neural networks, Proceedings of the 4th ICMLC, (2005) 2890–2894.

[38] X. Wang, N. Gao, S. Cai, M. Huang, An artificial fish swarm algorithm based and ABC supported QoS unicast routing scheme in NGI, Lecture Notes in Computer Science, 4331, ISPA 2006 G. Min et al.(eds.), Springer-Verlag, (2006) 205–214.

[39] Z.Y. Wu, F.S. Bai, H.W. Lee, Y.J. Yang, A filled function method for constrained global optimization, Journal of Global Optimization, 39 (2007) 495–507.

[40] E. Zahara, C.-H. Hu, Solving constrained optimization problems with hybrid particle swarm optimization, Engineering Optimization, 40 (2008) 1031–1049.

[41] W. Zhu, M.M. Ali, Solving nonlinearly constrained global optimization problem via an auxiliary function method, Journal of Computational and Applied Mathematics, 230 (2009) 491–503.